VIKRAM DEB AUTONOMOUS COLLEGE

JEYPORE, KORAPUT, ODISHA

COURSE OF STUDIES
OF

BACHELOR DEGREE SCIENCE UNDER CBCS

Subject: **COMPUTER SCIENCE**

FROM ADMISSION BATCH 2019 ONWARD

Published by

VIKRAM DEB AUTONOMOUS COLLEGE
JEYPORE, KORAPUT, ODISHA
Website: www.vikramdebcollege.ac.in

Preamble

Information and Communication Technology (ICT) has today become integral part of all industry domains as well as fields of academics and research. The industry equirements and technologies have been steadily and rapidly advancing. Organizations are ncreasingly opting for open source systems. The students too these days are thinking beyond career in the industry and aiming for research opportunities. A genuine attempt has been made while designing the new syllabus for this 3- year B. Sc. Computer Science (H) course. Not only does it prepare the students for a career in Software industry, it also motivates them towards further studies and research opportunities. The core philosophy of overall syllabus is to:

a. Form strong foundation of Computer science,

b. Introduce emerging trends to the students in gradual way,

c. Groom the students for the challenges of ICT industry

The Government of Odisha has initiated several measures to bring equity, efficiency and excellence in the Higher Education System of the State of Odisha in line with the University Grants Commission (UGC). The important measures taken to enhance academic standards and quality in higher education include innovation and improvements in curriculum, teaching learning process, examination and evaluation systems, besides governance and other matters. The Government of Odisha has formulated various regulations and guidelines from time to time to improve the higher education system and maintain minimum standards and quality across the Universities & Colleges in Odisha in line with UGC. The academic reforms recommended by the UGC in the recent past have led to overall improvement in the higher education system.

However, due to lot of diversity in the system of higher education, there are multiple approaches followed by universities towards examination, evaluation and grading system. While the Universities and Colleges must have the flexibility and freedom in designing the examination and evaluation methods that best fits the curriculum, syllabi and teaching–learning methods, there is a need to devise a sensible system for awarding the grades based on the performance of students. Presently the performance of the students is reported using the conventional system of marks secured in the examinations or grades or both. The conversion from marks to letter grades and the letter grades used vary widely across the Universities and Colleges in the states as well as the country. This creates difficulty for the academia and the employers to understand and infer the performance of the students graduating from different universities and colleges based on grades. The grading system is considered to be better than the conventional marks system and hence it has been followed in the top institutions in India and abroad. So, it is desirable to introduce uniform grading system. This will facilitate student mobility across institutions within and across countries and also enable potential employers to assess the performance of students. To bring in the desired uniformity, in grading system and method for computing the cumulative grade point average (CGPA) based on the performance of students in the examinations, the UGC has formulated these guidelines, which is being adopted by the state of Odisha.

CHOICE BASED CREDIT SYSTEM (CBCS): The CBCS provides an opportunity for the students to choose courses from the prescribed courses comprising core, elective/minor or skill based courses. The courses can be evaluated following the grading system, which is considered to be better than the conventional marks system. Therefore, it is necessary to introduce uniform grading system in the entire higher education in Odisha. This will benefit the students to move across institutions within Odisha to begin with and across states and countries. The uniform grading system will also enable potential employers in assessing the performance of the candidates. In order to bring uniformity in evaluation system and computation of the Cumulative Grade Point Average (CGPA) based on student's performance in examinations, the UGC has formulated the guidelines to be followed. Outline of Choice Based Credit System:

1. **Core Course**: A course, which should compulsorily be studied by a candidate as a core requirement is termed as a Core course.

2. Elective Course: Generally, a course which can be chosen from a pool of courses and which may be very specific or specialized or advanced or supportive to the discipline/ subject of study or which provides an extended scope or which enables an exposure to some other discipline/subject/domain or nurtures the candidate's proficiency/skill is called an Elective Course.

2.1 Discipline Specific Elective (DSE) Course: Elective courses may be offered by the main discipline/subject of study is referred to as Discipline Specific Elective. The University/Institute may also offer discipline related Elective courses of interdisciplinary nature (to be offered by main discipline/subject of study).

2.2 Dissertation/Project: An elective course designed to acquire special/advanced knowledge, such as supplement study/support study to a project work, and a candidate studies such a course on his own with an advisory support by a teacher/faculty member is called dissertation/project.

2.3 Generic Elective (GE) Course: An elective course chosen generally from an unrelated discipline/subject, with an intention to seek exposure is called a Generic Elective.

P.S.: A core course offered in a discipline/subject may be treated as an elective by other discipline/subject and vice versa and such electives may also be referred to as Generic Elective.

3. **Ability Enhancement Courses (AEC)/Competency Improvement Courses/Skill Development Courses/Foundation Course**: They ((i) Environmental Science, (ii) English/MIL Communication) are mandatory for all disciplines. AEC courses are value-based and/or skill based and are aimed at providing hands-on-training, competencies, skills, etc.

Project work/Dissertation is considered as a special course involving application of knowledge in solving / analyzing /exploring a real life situation / difficult problem. A Project/Dissertation work would be of 6 credits. A Project/Dissertation work may be given in lieu of a discipline specific elective paper.

GUIDELINES FOR PROJECT FORMULATION

As the project work constitutes a major component in most of the professional programs and it is to be carried out with due care and should be executed with seriousness by the candidates.

TYPE OF PROJECT

As majority of the students are expected to work out a real-life project in some industry/research and development laboratories/educational institutions/software companies, it is suggested that the project is to be chosen which should have some direct relevance in day-to-day activities of the candidates in his/her institution. It is not mandatory for a student to work on a real-life project. The student can formulate a project problem with the help of Guide.

PROJECT PROPOSAL (SYNOPSIS)

The project proposal should be prepared in consultation with the guide. The project proposal should clearly state the project objectives and the environment of the proposed project to be undertaken. The project work should compulsorily include the software development. The project proposal should contain complete details in the following form:

1. Title of the Project
2. Introduction and Objectives of the Project

3. Project Category (RDBMS/OOPS/Networking/Multimedia/Artificial Intelligence/Expert Systems etc.)

4. Analysis (DFDs at least up to second level, ER Diagrams/ Class Diagrams/ Database Design etc. as per the project requirements).

5. A complete structure which includes: Number of modules and their description to provide an estimation of the student's effort on the project. Data Structures as per the project requirements for all the modules. Process Logic of each module. Testing process to be used. Reports generation

6. Tools / Platform, Hardware and Software Requirement specifications

7. Future scope and further enhancement of the project.

Programme Outcomes Programme Specific Outcomes and Course Learning Outcomes

1. **Programme Outcomes (PO)**

   Students in the undergraduate program in Computer Science at Vikram Deb Autonomous College, Jeypore (K) under Berhampur University are expected to master the following competencies.

   PO1: Knowledge of mathematics, science and subjective computer fundamentals as specialization to the solution of complex software engineering problems.

   PO2: A thorough understanding of and ability to apply the core principles and practices of computing.

   PO3: Identify, formulate, review research literature and analyze computer engineering problems.

   PO4: Design and development of software components or process that meets the specific need.

   PO5: Create, select and apply appropriate IT tools, techniques, resources and modern engineering principles.

   PO6: Apply reasoning informed by contextual knowledge to assess societal, environmental, healthy, safety, legal and cultural issues and consequent responsibilities relevant to the professional engineering practice.

   PO7: Apply ethical principles and commit to professional and social ethics.

   PO8: Communicate effectively in a variety of professional contexts, design documents and make effective presentations.

   PO9: Demonstrate knowledge and understanding of the engineering and management principles as a member and leader in a team and to manage projects.

2. **Program Specific Outcomes (PSO)**

   On completion of the B.Sc. (Computer Science-Honours) degree the graduates will be able to:

   PSO1: Apply standard Software engineering practices and strategies in real-time software project development using open source programming environment or commercial environment to deliver quality product for the organization.

PSO2: Design and develop computer programmes or computer based systems in the areas related to algorithms, networking, web design, cloud computing, artificial intelligence and data engineering.

PSO3: Acquaint with the contemporary trends in industrial or research basis and thereby innovate novel solutions to existing problems.

3. Course Learning Outcomes (CO)

By the time of graduation for the degree of Bachelor of Science Honours in Computer Science, students will have the ability to:

CO1: Learn both theory and practical aspect on each subject.

CO2: Demonstrate the knowledge of Computer Science and Computer programming based problem solving skills.

CO3: Ability Enhancement Compulsory Course (**AECC**) provides ability to students to communicate effectively.

CO4: Students able to understand basic fundamental concepts on Core Course(**CC**) like programming language, algorithms, data structure, database concepts, networking, operating systems, computer graphics, web designing, software engineering and on other cutting-edge technologies like artificial intelligence.

CO5: Skill Enhancement Compulsory Course (**SEC**) provides ability to learn and communicate effectively simultaneous increase quantitative aptitude and reasoning capability.

CO6: Discipline Specific Elective (**DSE**) courses such as Numerical techniques, Unix Operating Systems, Data Science offer emerging technology knowledge that are being used in current science and technologies.

CO7: Ability to pursue specialized higher studies and to take up employment in the IT industry.

CO8: Ability to operate, manage, deploy, configure computer network, hardware, software operation of an organization.

CO9: Design and develop computer programs/computer-based systems in the areas related to algorithms, networking, web design, artificial intelligence, IoT and data analytics.

Curriculum Structure

Semester	Core Courses (CC)	Elective Courses				Semester wise credits
		DSE	GE	AEC	SEC	
I	CC 1, CC 2	-	GE 1	AEC 1		22
II	CC 3, CC 4	-	GE 2	AEC 2		22
III	CC 5, CC 6, CC 7	-	GE 3	-	SEC 1	26
IV	CC 8, CC 9, CC 10	-	GE 4	-	SEC 2	26
V	CC 11, CC 12	DSE 1 DSE 2	-	-	-	24
VI	CC 13, CC 14	DSE 3 DSE 4	-	-	-	24
Total Minimum Credits	84	24	24	08	04	144

Mark distribution or Evaluation Scheme

1. For Practical Course

Mid-Sem	End-Sem	End-Sem (Practical)
15	60	25

2. For Non-Practical Course

Mid-Sem	End-Sem
20	80

3. For Project

Project Report	Presentation	Viva-Voce
60	20	20

Detailed Curriculum

First Semester

Course Opted	Course Name	Credits	Marks
Ability Enhancement Course-1	AEC-1 (Environmental Science)	4	100
Core Course-1	Programming using C	4	75
Core Course-1 Practical	Programming using C LAB	2	25
Core Course-2	Digital Logic	4	75
Core Course-2 Practical	Digital Logic Lab	2	25
Generic Elective-1	GE-1	4	75
Generic Elective-1 Practical	GE-1 Tutorial/ LAB	2	25

Total Credit- 22 **Total Marks- 400**

Second Semester

Course Opted	Course Name	Credits	Marks
Ability Enhancement Course-2	AEC-2 (English Communication/MIL)	4	100
Core Course-3	Programming using C++	4	75
Core Course-3 Practical	Programming using C++ LAB	2	25
Core Course-4	Data Structures	4	75
Core Course-4 Practical	Data Structures LAB	2	25
Generic Elective-2	GE-2	4	75
Generic Elective-2 Practical	GE-2 Tutorial/ LAB	2	25

Total Credit- 22 **Total Marks- 400**

Third Semester

Course Opted	Course Name	Credits	Marks
Core Course-5	JAVA Programming	4	75
Core Course-5	Practical JAVA Programming LAB	2	25
Core Course-6	Database Systems	4	75
Core Course-6 Practical	Database Systems LAB	2	25
Core Course-7	Discrete Mathematical Structures	4	75
Core Course-7	Practical Discrete Mathematical Structures LAB	2	25
Skill Enhancement Course-1	SEC-1	2	50
Generic Elective-3	GE-3	4	75
General Elective-3 Practical	GE-3 Tutorial/ LAB	2	25

Total Credit- 26 **Total Marks- 450**

Fourth Semester

Course Opted	**Course Name**	**Credits**	**Marks**
Core Course-8	Operating Systems	4	75
Core Course-8 Practical	Operating Systems LAB	2	25
Core Course-9	Computer Networks	4	75
Core Course-9 Practical	Computer Networks LAB	2	25
Core Course-10	Computer Graphics	4	75
Core Course-10 Practical	Computer Graphics LAB	2	25
Skill Enhancement Course-2	SEC-2	2	50
Generic Elective-4	GE-4	4	75
General Elective-4 Practical	GE-4 Tutorial/ LAB	2	25

Total Credit- 26 **Total Marks- 450**

Fifth Semester

Course Opted	**Course Name**	**Credits**	**Marks**
Core Course-11	Web Technology	4	75
Core Course-11 Practical	Web Technology LAB	2	25
Core Course-12	Software Engineering	4	75
Core Course-12 Practical	Software Engineering Lab	2	25
Discipline Specific Elective-1	DSE-1	4	75
Discipline Specific Elective-1	Practical DSE-1 LAB/ Tutorial	2	25
Discipline Specific Elective-2	DSE-2	4	75
Discipline Specific Elective-2	Practical DSE-2 LAB/ Tutorial	2	25

Total Credit- 24 **Total Marks- 400**

Sixth Semester

Course Opted	**Course Name**	**Credits**	**Marks**
Core Course-13	Artificial Intelligence	4	75
Core Course-13	Practical Artificial Intelligence LAB	2	25
Core Course-14	Algorithm Design Techniques	4	75
Core Course-14	Practical Algorithm Design Techniques LAB	2	25
Discipline Specific Elective-3	DSE-3	4	75
Discipline Specific Elective-3	Practical DSE-3 LAB/ Tutorial	2	25

Discipline Specific Elective-4	DSE-4	4	75
Discipline Specific Elective-4	Practical DSE-4 LAB/ Tutorial	2	25

Total Credit- 24 **Total Marks- 400**

Semester – 1st to 6th Semester	
Total Credit- 144	**Total Marks- 2500**

CORE Papers: (Credit: 06 each)

CORE – 1: Programming Using C

CORE – 2: Digital Logic

CORE – 3: Programming Using C++

CORE – 4: Data Structure

CORE – 5: Java Programming

CORE – 6: Database Systems

CORE – 7: Discrete Mathematical Structures

CORE – 8: Operating System

CORE – 9: Computer Network

CORE – 10: Computer Graphics

CORE – 11: Web Technologies

CORE – 12: Software Engineering

CORE – 13: Artificial Intelligence

CORE – 14: Algorithm Design Techniques

Discipline Specific Electives (DSE) Papers:

DSE–1: Numerical Techniques

DSE–2: Unix Shell Programming

DSE–3: Data Science

DSE–4: Project Work / Dissertation

Skill Enhancement Courses (SEC):

SEC – 1: Communicative English

SEC – 2: Quantitative Aptitude.

Ability Enhancement Courses (AEC):

AEC – 1: Environmental Science.

AEC – 2: English Communication/MIL.

DETAILED SYLLABUS

CORE – 1: Programming Using C

OBJECTIVES:

- To learn basics of C programming language.
- To be able to develop logics to create programs/ applications in C.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Understanding a functional hierarchical code organization.
- Ability to define and manage data structures based on problem subject domain.
- Ability to work with textual information, characters and strings.
- Ability to work with arrays of complex objects.
- Understanding a concept of object thinking within the framework of functional model.
- Understanding a concept of functional hierarchical code organization.
- Understanding a defensive programming concept. Ability to handle possible errors during program execution.

Unit-1

Introduction: Introduction to Programming Language, Introduction to C Programming, Keywords & Identifiers, Constants, Variables, Input and Output Operations, Compilation and pre-processing, **Data types**: Different data types, Data types qualifier, modifiers, Memory representation, size and range, **Operators:** Operators (Arithmetic, Relational, Logical, Bitwise, Assignment & compound assignment, Increment & Decrement, Conditional), Operator types (unary, binary, ternary). Expressions, Order of expression (Precedence and associativity)

Control structures: Decision Making and Branching (Simple IF Statement, IF…ELSE Statement, Nesting IF… ELSE Statement, ELSE IF Ladder), Selection control structure (Switch Statement).

Unit-2

Loops: The WHILE Statement, The DO…WHILE Statement, The FOR Statement, Jumps in Loops, **Array**: Concept of Array, Array Declaration, types of array (one and multiple dimension), Character Arrays and Strings, Subscript and pointer representation of array, Array of Pointers, Limitation of array, **Pointers**: Concept of Pointer (null pointer, wild pointer, dangling pointer, generic pointer), Pointer

Expressions, Accessing the Address of a Variable, Declaring Pointer Variables, Initializations of Pointer Variable, Accessing a Variable through its Pointer, Pointer arithmetic.

Unit-3

Storage class: Types (auto, register, static, extern), scope rules, declaration and definition. **Function**: Function & types (User defined function, library function) Function Definition, Declaration, Function Calls, Header file and library, Function Arguments, string handling function (strlen, strcmp, strcpy, strncpy, strcat, strstr), Function recursion, Functions Returning Pointers, Pointers to Functions, Command line arguments, Application of pointer (dynamic memory allocation).

Unit-4

Structure and Union: Defining, Declaring, Accessing, Initialization Structure, nested structure, self-referential structure, bit-field, Arrays of Structures, Structures and Functions, Unions, difference between structure and union, active data member, structure within union, Self-referential Structure, **File**: File Management in C, Defining and Opening a File, File opening modes (read, write, append), Closing a File, File operations, file and stream, Error Handling During I/O Operations, sequential and random access file, low level and high level file.

Text Books:

1. E. Balagurusamy, “Programming in ANSI C”, 4/e, (TMH)

Reference Books:

1. B. Kernighan & Dennis Ritchie, “The C Programming Language”, 2/e PHI
2. Paul Deitel, Harvey Deitel, “C: How to Program”, 8/e, Prentice Hall.
3. P.C. Sethi, P.K. Behera, “Programming using C”, Kalyani Publisher, Ludhiana

Core-1 Practical: Programming Fundamentals using C Lab

1. Write a Program to find greatest among three numbers.
2. Write a Program to all arithmetic operation using switch case.
3. Write a Program to print the sum and product of digits of an integer.
4. Write a Program to reverse a number.
5. Write a Program to compute the sum of the first n terms of the following series
S = 1+1/2+1/3+1/4+……
6. Write a Program to compute the sum of the first n terms of the following series
S =1-2+3-4+5……………
7. Write a function that checks whether a given string is Palindrome or not. Use this function to find whether the string entered by user is Palindrome or not.

8. Write a function to find whether a given no. is prime or not. Use the same to generate the prime numbers less than 100.
9. Write a Program to compute the factors of a given number.
10. Write a program to swap two numbers using macro.
11. Write a Program to print a triangle of stars as follows (take number of lines from user):
12. Write a Program to perform following actions on an array entered by the user:
a) Print the even-valued elements
b) Print the odd-valued elements
c) Calculate and print the sum and average of the elements of array
d) Print the maximum and minimum element of array
e) Remove the duplicates from the array
f) Print the array in reverse order
The program should present a menu to the user and ask for one of the options. The menu should also include options to re-enter array and to quit the program.

13. Write a Program that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.

14. Write a program that swaps two numbers using pointers.

15. Write a program in which a function is passed address of two variables and then alter its contents.

16. Write a program which takes the radius of a circle as input from the user, passes it to another function that computes the area and the circumference of the circle and displays the value of area and circumference from the main( ) function.

17. Write a program to find sum and average of n elements entered by the user. To write this program, allocate memory dynamically using malloc( ) / calloc( ) functions.

18. Write a menu driven program to perform following operations on strings:

a) Show address of each character in string

b) Concatenate two strings without using strcat function.

c) Concatenate two strings using strcat function.

d) Compare two strings

e) Calculate length of the string (use pointers)

f) Convert all lowercase characters to uppercase

g) Convert all uppercase characters to lowercase

h) Calculate number of vowels

i) Reverse the string

19. Given two ordered arrays of integers, write a program to merge the two-arrays to get an ordered array.

20. Write a program to copy the content of one file to other.

CORE–2: DIGITAL LOGIC

OBJECTIVES

- To understand different methods used for the simplification of Boolean functions and binary arithmetic. It covers topics such as binary addition and subtraction, flip-flops
- To design and implement combinational circuits, synchronous & asynchronous
- sequential circuits.
- To study in detail about Semiconductor Memory Systems.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Digital Logic facilitates computing, robotics and other electronic applications.
- Digital Logic Design is foundational to the fields of electrical engineering and computer engineering.
- Digital Logic designers build complex electronic components that use both electrical and computational characteristics.

Unit-1

Character Codes, Decimal System, Binary System, Decimal to Binary Conversion, Hexadecimal Notation, Boolean Algebra, Basic Logic Functions: Electronic Logic Gates, Synthesis of Logic Functions, Minimization of Logic Expressions, Minimization using Karnaugh Maps, Synthesis with NAND and NOR Gates, Tri-State Buffers

Unit-2

Arithmetic: Addition and Subtraction of Signed Numbers, Addition/ Subtraction Logic Unit, Design of Fast Adders: Carry-Lookahead Addition, Multiplication of Positive Numbers, Signed- Operand Multiplication: Booth Algorithm, Fast Multiplication: Bit-Pair Recodng Multipliers,

Carry-Save Addition of Summands, Integer Division, Floating-Point Numbers and Operations: IEEE Standard for Floating-Point Numbers, Arithmetic Operations on Floating-Point Numbers, Guard Bits and Truncation, Implementing Floating-Point Operations.

Unit-3

Flip-Flops, Gated Latches, Master-Slave Flip-Flops, Edge-Triggering, T Flip-Flops, JK Flip-Flops. Registers and Shift Registers, Counters, Decoders, Multiplexers, Programmable Logic Devices (PLDs), Programmable Array Logic (PAL), Complex Programmable Logic Devices

(CPLDs), Field-Programmable Gate Array (FPGA), Sequential Circuits, UP/ DOWN Counters, Timing Diagrams, The Finite State Machine Model, Synthesis of Finite State Machines.

Unit-4

Memory System: Semiconductor RAM Memories, Internal Organization of Memory Chips, Static Memories, Asynchronous DRAMS, Synchronous DRAMS, Structure of Large Memories, Memory System Considerations, RAMBUS Memory. Read-Only Memories: ROM, PROM, EPROM, EEPROM, Flash Memory, Speed, Size, and Cost of Memory. Secondary Storage: Magnetic Hard Disks, Optical Disks, Magnetic Tape Systems.

Text Books:

1. Carl Hamacher, Z. Vranesic, S. Zaky: Computer Organization, 5/e (TMH)

Reference Books:

1. M. Morris Mano: Digital Logic and Computer Design, Pearson

CORE–2 Practical: Digital Logic Lab

1. Introduction to Xilinx software (VHDL)

Write the VHDL code for

2. Realizing all logic gates.
3. Combination Circuit.
4. ADDER.
5. SUBTRACTOR.
6. MUX.
7. DE-MUX.
8. Encoder.
9. Decoder.
10. PAL.
11. PLA.

Write the VHDL program for the following Sequential Logic Circuits

12. Flip Flops.
13. Shift Registers.
14. Counters.
15. Memory Elements.

CORE–3: Programming Using C++

OBJECTIVES

- To know about the Object Oriented Programming concepts.
- To learn basics of C++ programming language.
- To be able to develop logics to create programs/ applications in C++.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Understand the difference between the top-down and bottom-up approach
- Describe the object-oriented programming approach in connectionWith C++
- Apply the concepts of object-oriented programming
- Illustrate the process of data file manipulations using C++
- Apply virtual and pure virtual function & complex programming situation.

Unit-1

Principles of Object-Oriented Programming: Object-Oriented Programming (OOP) Paradigm, Basic Concepts of OOP, Benefits of OOP, Characteristics of OOPS, Object Oriented Languages, Applications of OOP. Introduction to C++, Difference between C & C++, Tokens, Data types, Operators, Structure of C++ Program, C++ statements, Expressions and Control Structures. Functions in C++: Argument passing in function, Inline Functions, Default Arguments, Const.
Arguments, Friend function.

Unit-2

Classes and Objects: Defining Member Functions, Making an outside Function Inline, Nested Member Functions, Private Member Functions, Arrays within a Class, Memory Allocation for Objects, Static Data Members, Static Member Functions, Arrays of Objects, Objects as Function Arguments, Friend Functions.
Constructors & Destructors: Constructors, Parameterized Constructors, Constructors with Default Arguments, Dynamic Initialization of Objects, Copy Constructor, Dynamic Constructors, Destructors.

Unit-3

Inheritance: Basics of Inheritance, Type of Inheritance, Virtual Base Classes, Abstract Classes, Member Classes, Nesting of Classes. Polymorphism: Pointers, Pointers to Objects, this Pointer, Pointers to Derived Classes, Virtual Functions, Pure Virtual Functions, Function Overloading, Operator Overloading.

Unit-4

Managing Console I/O Operations: C++ Streams, C++ Stream Classes, Unformatted I/O Operations, Formatted Console I/O Operations, Managing Output with Manipulators. Files: Classes for File Stream Operations, Opening and Closing a File, Detecting end-of-file, File Modes, File Pointers and their Manipulations, Sequential Input and Output Operations, Updating a File: Random Access, Error Handling during File Operations, Command-line Arguments.

Text Books

1. E. Balgurusawmy, Object Oriented Programming with C++, 4/e (TMH).
2. Paul Deitel, Harvey Deitel, "C++: How to Program", 9/e. Prentice Hall.

Reference Books:

1. Bjarne Stroustroup, Programming - Principles and Practice using C++, 2/e, Addison-Wesley 2014
2. Herbtz Schildt, C++: The Complete reference, MGH, 4/ed.
3. P. C. Sethi, P. K. Behera, “Programming in C++”- Kalyani Publisher, Ludhiana

CORE–3 Practical: Programming using C++ Lab

1. Write a Program to find greatest among three numbers using nested if…else statement.
2. Write a Program to check a number is prime or not.
3. Write a Program to find the GCD and LCM of two numbers.
4. Write a program to print the result for following series: 1! + 2! + 3! + …………
5. Write a program to print multiplication table from 1 to 10.
6. Write a Program for Swapping of two numbers using pass by value.
7. Write a Program for Swapping of two numbers using pass by address.
8. Write a Program for Swapping of two numbers using pass by reference.
9. Write a Program to find sum of four numbers using default argument passing.
10. Write a Program to find square and cube of a number using inline function.
11. Write a Program to find the factorial of a number.
12. Write a Program to find reverse of a number.
13. Write a program to find sum of four numbers using default argument passing in member function.
14. Write a Program to find area of circle, triangle and rectangle using function overloading.
15. Write a program to distinguish the properties of static and non-static ata members.
16. Write a program to show the method of accessing static private member function.
17. Write a program to show the ways of calling constructors and destructors.
18. Write a program to perform ++ operator overloading using member function.
19. Write a program to perform ++ operator overloading using friend function.
20. Write a program to perform + operator overloading for two complex number addition.
21. Write a program to perform + operator overloading for string concatenation.
22. Write a program to perform single inheritance.
23. Write a program to perform multiple inheritance.
24. Write a program to create an integer array using new operator and find the sum and average of array elements.
25. Write a program to implement virtual destructor.

26. Create the Person class. Create some objects of this class (by taking information from the user). Inherit the class Person to create two classes Teacher and Student class. Maintain the respective information in the classes and create, display and delete objects of these two classes (Use Runtime Polymorphism).
27. Write a program to Copy the contents of one file to other.

CORE–4: Data Structure

OBJECTIVES

- To learn how the choice of data structures impacts the performance of programs.
- To study specific data structures such as arrays, linear lists, stacks, queues, hash tables, binary trees, binary search trees, heaps and AVL trees.
- To learn efficient searching and sorting techniques.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Ability to program data structures and use them in implementations of abstract data types.
- Ability to devise novel solutions to small scale programming challenges involving data structures and recursion.
- Understanding of basic algorithmic complexity.

Unit-1

Introduction: Basic Terminology, Data structure, Time and space complexity, Review of Array, Structures, Pointers.

Linked Lists: Dynamic memory allocation, representation, Linked list insertion and deletion, Searching, Traversing in a list, Doubly linked list, Sparse matrices.

Unit-2

Stack: Definition, Representation, Stack operations, Applications (Infix–Prefix–Postfix Conversion & Evaluation, Recursion).

Queues: Definition, Representation, Types of queue, Queue operations, Applications.

Unit-3

Trees: Tree Terminologies, General Tree, Binary Tree, Representations, Traversing, BST, Operations on BST, Heap tree, AVL Search Trees, M-way search tree, Applications of all trees.

Unit-4

Sorting: Exchange sorts, Selection Sort, Bubble sort, Insertion Sorts, Merge Sort, Quick Sort, Radix Sort, Heap sort.

Searching: Linear search, Binary search.

Text book

1. Classic Data Structure , D. Samanta , PHI , 2/ed.

REFERENCES

1. Ellis Horowitz, Sartaj Sahni, “Fundamentals of Data Structures”, Galgotia Publications, 2000.

2. Sastry C.V., Nayak R, Ch. Rajaramesh, Data Structure & Algorithms, I. K. International Publishing House Pvt. Ltd, New Delhi.

CORE – 4 Practical: Data Structure Lab

Write a C/ C++ Program for the followings

1. To insert and delete elements from appropriate position in an array.
2. To search an element and print the total time of occurrence in the array.
3. To delete all occurrence of an element in an array.
4. Array implementation of Stack.
5. Array implementation of Linear Queue.
6. Array implementation of Circular Queue.
7. To implement linear linked list and perform different operation such as node insert and delete, search of an item, reverse the list.
8. To implement circular linked list and perform different operation such as node insert and delete.
9. To implement double linked list and perform different operation such as node insert and delete.
10. Linked list implementation of Stack.
11. Linked list implementation of Queue.
12. Polynomial representation using linked list.
13. To implement a Binary Search Tree.
14. To represent a Sparse Matrix.
15. To perform binary search operation.
16. To perform Bubble sort.
17. To perform Selection sort.
18. To perform Insertion sort.
19. To perform Quick sort.
20. To perform Merge sort.

CORE – 5: Operating System

OBJECTIVES

- To understand Operating system structure and services.
- To understand the concept of a Process, memory, storage and I/O management.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Students will learn how Operating System is Important for Computer System.
- To make aware of different types of Operating System and their services.
- To learn different process scheduling algorithms and synchronization techniques to achieve better performance of a computer system.
- To know virtual memory concepts.
- To learn secondary memory management.

Unit–1

Introduction to Operating System, System Structures: Operating system services, system calls, system programs, Operating system design and implementation, Operating system structure.

Unit–2

Process Management: Process Concept, Operations on processes, Process scheduling and algorithms, Inter-process Communication, Concepts on Thread and Process, Deadlocks: Deadlock detection, deadlock prevention, and deadlock avoidance fundamentals.

Unit-3

Memory Management Strategies: Swapping, Contiguous Memory Allocation, Paging, Segmentation, Virtual Memory Management: Concepts, implementation (Demand Paging), Page Replacement, Thrashing.

Unit–4

Storage Management: File System concept, Access Methods, File System Mounting, File

Sharing and File Protection, Implementing File Systems, Kernel I/O Systems.

Text book – Operating System Concepts, Abraham Silberschatz, Peter B. Galvin, and Greg Gagne, Eighth Edition, Wiley Student Edition 2009.

Reference book:

1. Morden Operating System , Tanenbaum ,Pearson , 4/ed. 2014

2. Richard F Ashley, Linux with Operating System Concepts, Chapman and Hall/CRC Published August 26, 2014

3. Richard Blum, Linux Command Line and Shell Scripting Bible, O' Reilly

CORE-5 Practical: Operating System Lab

1. Write a program (using *fork()* and/or *exec()* commands) where parent and child execute:

a) same program, same code.

b) same program, different code.

c) before terminating, the parent waits for the child to finish its task.

2. Write a program to report behavior of Linux kernel including kernel version, CPU type and model. (CPU information)

3. Write a program to report behavior of Linux kernel including information on configured memory, amount of free and used memory. (memory information)

4. Write a program to print file details including owner access permissions, file access time, where file name is given as argument.

5. Write a program to copy files using system calls.

6. Write a program using C to implement FCFS scheduling algorithm.

7. Write a program using C to implement Round Robin scheduling algorithm.

8. Write a program using C to implement SJF scheduling algorithm.

9. Write a program using C to implement non-preemptive priority based scheduling algorithm.

10. Write a program using C to implement preemptive priority based scheduling algorithm.

11. Write a program using C to implement SRTF scheduling algorithm.

21

12. Write a program using C to implement first-fit, best-fit and worst-fit allocation strategies.

CORE-6: Database Systems

OBJECTIVES

- To learn the fundamental elements of database system.
- To learn the basic concepts of relational database management systems.
- To learn various SQL commands.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Describe the fundamental elements of relational database management systems.
- Explain the basic concepts of relational data model, entity-relationship model, relational database design, relational algebra and SQL.
- Design ER-models to represent simple database application scenarios
- Convert the ER-model to relational tables, populate relational database and formulate SQL queries on data.
- Improve the database design by normalization.
- Familiar with basic database storage structures and access techniques: file and page
- Organizations, indexing methods including B tree, and hashing.

Unit-1

Introduction to Database and Database Users, Database System Concepts and Architecture: data Models, schema, and instances, Conceptual Modeling and Database Design: Entity Relationship (ER) Model: Entity Types, Entity Sets, Attributes, Keys, Relationship Types, Relationship Sets, Roles and Structural Constraints, Weak Entity Types, ER Naming Conventions. Enhanced Entity-Relationship (EER) Model.

Unit-2

Database Design Theory and Normalization: Functional Dependencies, Normal Forms based on Primary Keys, Second and third Normal Forms, Boyce-Codd Normal Form, Multivalued Dependency and Fourth Normal Form, Join Dependencies and Fifth Normal Form.

Unit-3

Relational data Model and SQL: Relational Model Concepts, Basic SQLs, SQL Data Definition and Data types, Constraints in SQL, Retrieval Queries in SQL, INSERT, DELETE, UPDATE Statements in SQL, Relational Algebra and Relational Calculus: Unary Relational Operations: SELECT and PROJECT, Binary Relation: JOIN and DIVISION.

Unit-4

Introduction to Transaction Processing Concepts and Theory: Introduction to Transaction

Processing, Transaction and System Concepts, Properties of Transactions, Recoverability, Serializability, Concurrency Control Techniques, Locking techniques for Concurrency Control, Concurrency Control based on Time-Stamp Ordering.

Text Book:

1. Fundamentals of Database Systems, 6th edition, Ramez Elmasri, Shamkant B. Navathe, Pearson Education

Reference Book:

1. An Introduction to Database System, Date C. J. - Pearson Education, New Delhi - 2005

CORE-6 Practical: Database Systems Labs

Create and use the following database schema to answer the given queries.

EMPLOYEE Schema

Field Type NULL KEY DEFAULT

Eno Char(3) NO PRI NIL

Ename Varchar(50) NO NIL

Job_type Varchar(50) NO NIL

Manager Char(3) Yes FK NIL

Hire_date Date NO NIL

Dno Integer YES FK NIL

Commission Decimal(10,2) YES NIL

Salary Decimal(7,2) NO NIL

DEPARTMENT Schema

Field Type NULL KEY DEFAULT

Dno Integer No PRI NULL

Dname Varchar(50) Yes NULL

Location Varchar(50) Yes New Delhi

Query List

1. Query to display Employee Name, Job, Hire Date, Employee Number; for each employee
with the Employee Number appearing first.
2. Query to display unique Jobs from the Employee Table.
3. Query to display the Employee Name concatenated by a Job separated by a comma.
4. Query to display all the data from the Employee Table. Separate each Column by a comma
and name the said column as THE_OUTPUT.
5. Query to display the Employee Name and Salary of all the employees earning more than
$2850.
6. Query to display Employee Name and Department Number for the Employee No= 7900.
7. Query to display Employee Name and Salary for all employees whose salary is not in the range of $1500 and $2850.
8. Query to display Employee Name and Department No. of all the employees in Dept 10 and Dept 30 in the alphabetical order by name.
9. Query to display Name and Hire Date of every Employee who was hired in 1981.
10. Query to display Name and Job of all employees who don't have a current Manager.
11. Query to display the Name, Salary and Commission for all the employees who earn commission.
12. Sort the data in descending order of Salary and Commission.
13. Query to display Name of all the employees where the third letter of their name is 'A'.
14. Query to display Name of all employees either have two 'R's or have two 'A's in their name and are either in Dept No = 30 or their Mangers Employee No = 7788.
15. Query to display Name, Salary and Commission for all employees whose Commission Amount is 14 greater than their Salary increased by 5%.
16. Query to display the Current Date.
17. Query to display Name, Hire Date and Salary Review Date which is the 1st Monday after six months of employment.
18. Query to display Name and calculate the number of months between today and the date each employee was hired.
19. Query to display the following for each employee <E-Name> earns < Salary> monthly but wants <3*Current Salary>. Label the Column as Dream Salary.
20. Query to display Name with the 1st letter capitalized and all other letter lower case and

length of their name of all the employees whose name starts with ‘J’, ‘A’ and ‘M’.

21. Query to display Name, Hire Date and Day of the week on which the employee started.

22. Query to display Name, Department Name and Department No for all the employees.

23. Query to display Unique Listing of all Jobs that are in Department # 30.

24. Query to display Name, Department Name of all employees who have an ‘A’ in their name.

25. Query to display Name, Job, Department No. and Department Name for all the employees working at the Dallas location.

26. Query to display Name and Employee no. Along with their Manger’s Name and the Manager’s employee no; along with the Employees Name who do not have a Manager.

27. Query to display Name, Department No. And Salary of any employee whose department No. and salary matches both the department no. And the salary of any employee who earns a commission.

28. Query to display Name and Salaries represented by asterisks, where each asterisk (*) signifies $100.

29. Query to display the Highest, Lowest, Sum and Average Salaries of all the employees.

30. Query to display the number of employees performing the same Job type functions.

31. Query to display the no. of managers without listing their names.

32. Query to display the Department Name, Location Name, No. of Employees and the average salary for all employees in that department.

33. Query to display Name and Hire Date for all employees in the same dept. as Blake.

34. Query to display the Employee No. And Name for all employees who earn more than the average salary.

35. Query to display Employee Number and Name for all employees who work in a department with any employee whose name contains a ‘T’.

36. Query to display the names and salaries of all employees who report to King.

37. Query to display the department no, name and job for all employees in the Sales department.

CORE – 7: Discrete Mathematical Structure

OBJECTIVES

- To learn the mathematical foundations for Computer Science.
- Topics covered essential for understanding various courses.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Perform logical proofs.
- Apply recursive functions and solve recurrence relations.
- Determine equivalent logic expressions.
- Describe useful standard library functions, create functions, and declare parameters.
- Use graphs and trees.
- Apply basic and advanced principles of counting.
- Define sets and sequences.
- Calculate discrete probabilities.
- Design and evaluate Euler and Hamilton circuits.

Unit-1

Logics and Proof: Propositional Logic, Propositional Equivalences, Predicates and Quantifiers Nested Quantifiers, Rules inference, Mathematical Induction.

Sets and Functions: Sets, Relations, Functions, Closures of Equivalence Relations, Partial ordering well ordering, Lattice, Sum of products and product of sums principle of Inclusions and Exclusions

Unit-2

Combinatory: Permutations, Combinations, Pigeonhole principle

Recurrence Relation: Linear and Non-linear Recurrence Relations, Solving Recurrence Relation using Generating Functions.

Unit-3

Graphs: Introduction to graphs, graphs terminologies, Representation of graphs, Isomorphism,

Connectivity & Paths: Connectivity, Euler and Hamiltonian Paths, Introduction to tree, tree traversals, spanning tree and tree search: Breadth first search, Depth first search, cut-set, cutvertex.

Unit-4

Modeling Computation: Finite State Machine, Deterministic Finite Automata (DFA), Non- Deterministic Finite Automata (NFA), Grammars and Language, Application of Pumping Lemma for Regular Language.

Text Books:

1. "Discrete Mathematics and its Applications with Combinatory and Graph Theory" 7th edition by Kenneth H. Rosen.

Reference Books:

1. Elements of Discrete Mathematics by C. L. Liu and D.P. Mohapatra, TMH, 2012
2. J. P Tremblay, R. Manohar, "Discrete Mathematical Structures with Applications to Computer Science", TMH, 1997.
3. A Modern Approach to Discrete Mathematics and Structure by J. K. Mantri & T. K Tripathy ,Laxmi Publication

CORE – 7 Practical: Discrete Mathematical Structure Lab

Write the following programs using C/ C++

1. Tower of Hanoi
2. Graph representation using Adjacency List.
3. Graph representation using Adjacency Matrix.
4. String Matching using finite state machine.
5. Detecting whether a number is even or odd using Finite State Machine.
6. To identify keywords such as char, const, continue using Finite State Machine.
7. To find the power set for a given set.
8. To find GCD of two numbers using recursion.
9. To find Binomial coefficients.
10. To find Permutation and Combination result for a given pair of values n and r.
11. To check a number is prime or not.
12. To calculate the Euclidean distance between two points.
13. To find the Roots of polynomials.
14. Find the shortest path pair in a plane.

CORE–8: Java Programming

OBJECTIVES

- To learn the fundamentals of Object Oriented Programming in Java environment.
- To learn the use of Java language and the Java Virtual Machine.
- To write simple Java programming applications.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Use an integrated development environment to write, compile, run, and test simple object oriented Java programs.
- Read and make elementary modifications to Java programs that solve real-world problems.
- Validate input in a Java program.
- Identify and fix defects and common security issues in code.
- Document a Java program using Javadoc.
- Use a version control system to track source code in a project.

Unit-1

Introduction to Java: Java History, Architecture and Features, Understanding the semantic and syntax differences between C++ and Java, Compiling and Executing a Java Program, Variables, Constants, Keywords (super, this, final, abstract, static, extends, implements, interface) , Data Types, Wrapper class, Operators (Arithmetic, Logical and Bitwise) and Expressions, Comments, Doing Basic Program Output, Decision Making Constructs (conditional statements and loops) and Nesting, Java Methods (Defining, Scope, Passing and Returning Arguments,
Type Conversion and Type and Checking, Built-in Java Class Methods). Input through keyboard using Command line Argument, the Scanner class, Buffered Reader class.

Unit-2

Object-Oriented Programming Overview: Principles of Object-Oriented Programming, Defining & Using Classes, Class Variables & Methods, Objects, Object reference, Objects as parameters, final classes, Garbage Collection.
Constructor- types of constructor, this keyword, super keyword. Method overloading and Constructor overloading. Aggregation vs Inheritance, Inheritance: extends vs implements, types of Inheritance, Interface, Up-Casting, Down-Casting, Auto-

Boxing, Enumerations, Polymorphism, Method Overriding and restrictions. Package: Pre-defined packages and Custom packages.

Unit-3

Arrays: Creating & Using Arrays (1D, 2D, 3D and Jagged Array), Array of Object, Referencing Arrays Dynamically. Strings and I/O: Java Strings: The Java String class, Creating & Using String Objects, Manipulating Strings, String Immutability& Equality, Passing Strings To & From Methods, StringBuffer Classes and StringBuilder Classes. IO package: Understanding StreamsFile class and its methods, Creating, Reading, Writing using classes: Byte and Character streams, FileOutputStream, FileInputStream, FileWriter, FileReader, InputStreamReader, PrintStream, PrintWriter. Compressing and Uncompressing File.

Unit-4

Exception Handling, Threading, Networking and Database Connectivity: Exception types, uncaught exceptions, throw, built-in exceptions, Creating your own exceptions; Multi-threading: The Thread class and Runnable interface, creating single and multiple threads, Thread prioritization, synchronization and communication, suspending/resuming threads. Using java.net package, Overview of TCP/IP and Datagram programming. Accessing and manipulating databases using JDBC.

Text Books:

1. E. Balagurusamy, "Programming with Java", TMH, 4/Ed,

Reference books:

1. Herbert Schildt, "The Complete Reference to Java", TMH, 10/Ed.

CORE – 8 Practical: Java Programming Lab

1. To find the sum of any number of integers entered as command line arguments.
2. To find the factorial of a given number.
3. To convert a decimal to binary number.
4. To check if a number is prime or not, by taking the number as input from the keyboard.
5. To find the sum of any number of integers interactively, i.e., entering every number from the keyboard, whereas the total number of integers is given as a command line argument
6. Write a program that show working of different functions of String and String Buffer class like setCharAt( ), setLength( ), append( ), insert( ), concat( )and equals( ).

7. Write a program to create a – “distance” class with methods where distance is computed in terms of feet and inches, how to create objects of a class and to see the use of this pointer

8. Modify the – “distance” class by creating constructor for assigning values (feetandinches) to the distance object. Create another object and assign second object as reference variable to another object reference variable. Further create a third object which is a clone of the first object.

9. Write a program to show that during function overloading, if no matching argument is found, then Java will apply automatic type conversions (from lower to higher data type)

10. Write a program to show the difference between public and private access specifiers. The program should also show that primitive data types are passed by value and objects are passed by reference and to learn use of final keyword.

11. Write a program to show the use of static functions and to pass variable length arguments in a function.

14. Write a program to demonstrate the concept of boxing and unboxing.

15. Create a multi-file program where in one file a string message is taken as input from the user and the function to display the message on the screen is given in another file (make use of Scanner package in this program).

16. Write a program to create a multilevel package and also creates a reusable class to generate Fibonacci series, where the function to generate Fibonacci series is given in a different file belonging to the same package.

17. Write a program that creates illustrates different levels of protection in classes/subclasses belonging to same package or different packages

18. Write a program – “DivideByZero” that takes two numbers a and b as input, computes a/b, and invokes Arithmetic Exception to generate a message when the denominator is zero.

19. Write a program to show the use of nested try statements that emphasizes the sequence of checking for catch handler statements.

20. Write a program to create your own exception types to handle situation specific to your application (Hint: Define a subclass of Exception which itself is a subclass of Throwable).

21. Write a program to demonstrate priorities among multiple threads.

22. Write a program to demonstrate different mouse handling events like mouseClicked( ), mouseEntered(), mouseExited(), mousePressed(), mouseReleased( ) & mouseDragged().

23. Write a program to demonstrate different keyboard handling events.

CORE – 9: Computer Networks

OBJECTIVES

- To learn how do computers and terminals actually communicate with each other.
- To understand the parts of a communication network and how they work together.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Describe the general principles of data communication.
- Describe how computer networks are organized with the concept of layered approach.
- Describe how signals are used to transfer data between nodes.
- Implement a simple LAN with hubs, bridges and switches.
- Describe how packets in the Internet are delivered.
- Analyze the contents in a given data link layer
- Packet, based on the layer con-cept.
- Design logical sub-address blocks with a given address block.
- Decide routing entries given a simple example of network topology
- Describe what classless addressing scheme is.
- Describe how routing protocols work.

Unit-1

Introduction to Data Communications and Network Models: Protocols and Standards, Layers in OSI Models, Analog and Digital Signals, Transmission Modes, Transmission Impairment, Data Rate Limits, Performance, Digital Transmission, Network Devices & Drivers: Router, Modem, Repeater, Hub, Switch, Bridge (fundamental concepts only).

Unit-2

Signal Conversion: Digital-to-Digital Conversion, Analog-to-Digital Conversion, Digital-toanalog Conversion, Analog-to-analog Conversion. Transmission Media: Guided Media, Unguided Media, Switching Techniques: Packet Switching, Circuit Switching, Datagram Networks, Virtual-Circuit Networks, and Structure of a Switch.

Unit-3

Error Detection and Correction: Checksum, CRC, Data Link Control: Framing, Flow and Error Control, Noiseless Channels, Noisy channels, (Stop and Wait ARQ, Slidding Window Protocol , Go Back N, Selective Repeat) HDLC, Point-to-Point Protocol. Access Control: TDM, CSMA/CD, and Channelization (FDMA, TDMA, and CDMA).

Unit-4

Network Layer: Logical Addressing, IPv4 Addresses, IPv6 Addresses, Virtual-Circuit Networks: Frame Relay and ATM, Transport Layer: Process-Process Delivery: UDP, TCP. Application layers: DNS, SMTP, POP, FTP, HTTP, Basics of WiFi (Fundamental concepts only), Network Security: Authentication, Basics of Public Key and Private Key, Digital Signatures and Certificates (Fundamental concepts only).

Text Books:

1. Data Communications and Networking, Fourth Edition by Behrouza A. Forouzan,TMH.

Reference Books:

1. Computer Networks, A. S. Tanenbaum, 4th edition, Pearson Education.

CORE – 9 Practical: Computer Networks Lab

Use C/C++/ any Network Simulator

1. Simulate Even Parity generator and checker.
2. Simulate two dimensional Parity generator and checker.
3. Simulate checksum generator and checker.

22

4. Simulate Hamming code method.
5. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.
6. Simulate and implement stop and wait protocol for noisy channel.
7. Simulate and implement go back n sliding window protocol.
8. Simulate and implement selective repeat sliding window protocol.
9. Simulate and implement distance vector routing algorithm.

CORE – 10: Computer Graphics

OBJECTIVES

- To be able to learn the core concepts of Computer Graphics.
- To be able to create effective programs for solving graphics problems.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Explain the core concepts of computer graphics, including viewing. Projection, perspective, modelling and transformation in two and three dimensions.
- Apply the concepts of color models, lighting and shading models, textures, ray tracing, hidden surface elimination, anti-aliasing, and rendering.
- Interpret the mathematical foundation of the concepts of computer graphics.
- Describe the fundamentals of animation, parametric curves and surfaces, and spotlighting.
- Identify a typical graphics pipeline and apply graphics programming techniques to design and create computer graphics.
- Create effective C programs to solve graphics programming issues, including 3D transformation, objects modeling, color modeling, lighting, textures, and ray tracing.

Unit-1

Computer Graphics: A Survey of Computer graphics, Overview of Graphics System: Video Display Devices, Raster-Scan Systems, Input Devices, Hard-Copy Devices, Graphics Software.

Unit-2

Graphics Output Primitives: Point and Lines, Algorithms for line, circle & ellipse generation, Filled-Area Primitives. Attributes of Graphics Primitives: Point, line, curve attributes, fill area attributes, Fill methods for areas with irregular boundaries.

Unit-3

Geometric Transformations (both 2-D & 3-D): Basic Geometric Transformations, Transformation Matrix, Types of transformation in 2-D and 3-D Graphics: Scaling, Reflection, shear transformation, rotation, translation. 2-D, 3-D transformation using homogeneous coordinates.

Unit-4

Two Dimensional Viewing: Introduction to viewing and clipping, Viewing transformation in 2- D, Viewing pipeline, Clipping Window, Clipping Algorithms: Point clipping, Line clipping and Polygon clipping.

Text books

1. Mathematical Elements for Computer Graphics, D.F.Rogers & J. A. Adams, MGH, 2/ed.
2. Donald Hearn & M. Pauline Baker, “Computer Graphics with OpenGL”, Pearson Education.

Reference books

1. D. Hearn and M. Baker, “Computer Graphics with Open GL”, Pearson, 2/ed.
2. D. F. Rogers, “Procedural Elements for Computer Graphics”, MGH

CORE – 10 Practical: Computer Graphics Lab

Develop the programs using C/C++ or Java

1. Write a program to implement Bresenham’s line drawing algorithm.
2. Write a program to implement mid-point circle drawing algorithm.
3. Write a program to clip a line using Cohen and Sutherland line clipping algorithm.
4. Write a program to clip a polygon using Sutherland Hodgeman algorithm.
5. Write a program to fill a polygon using Scan line fill algorithm.
6. Write a program to apply various 2D transformations on a 2D object (use homogenous coordinates).
7. Write a program to apply various 3D transformations on a 3D object and then apply parallel and perspective projection on it.

CORE – 11: Web Technologies

OBJECTIVES

- To learn the fundamentals of web designing.
- To design and develop standard and interactive web pages.
- To learn some popular web scripting languages.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Analyze a web page and identify its elements and attributes.
- Create web pages using XHTML and Cascading Style Sheets (CSS).
- Build dynamic web pages using JavaScript (Client side programming). Create XML documents and Schemas.
- Build interactive web applications using HTML5, CSS, PHP.

Unit-1

Web Essentials: Clients, Servers and Communication: The Internet – Basic Internet protocols – The WWW, HTTP request message – response message, web clients web servers – case study. Introduction to HTML: HTML, HTML domains, basic structure of an HTML document – creating an HTML document, mark up tags, heading, paragraphs, line breaks, HTML tags. Elements of HTML, working with text, lists, tables and frames, working with hyperlink, images and multimedia, forms and controls

Unit-2

Introduction to cascading style sheets: Concepts of CSS, creating style sheet, CSS properties, CSS styling (background, text format, controlling fonts), working with the block elements and objects. Working who lists and tables, CSS ID and class. Box model (introduction, border properties, padding properties, margin properties), CSS colour, groping, Dimensions, display, positioning, floating, align, pseudo class, Navigation bar, image sprites.

Unit-3

Java scripts: Client side scripting, what is java script, simple java script, variables, functions, conditions, loops and repetitions. Java scripts and objects, java script own objects, the DOM and web browser environment, forms and validations. DHTML: Combining HTML, CSS, java scripts, events and buttons, controlling your browser.

Unit-4

PHP: Starting to script on server side, PHP basics, variables, data types, operators, expressions, constants, decisions and loop making decisions. Strings – creating, accessing strings, searching, replacing and formatting strings. Arrays: Creation, accessing array, multidimensional arrays, PHP with Database.

Text Book:

1. Web Technologies – Black Book – DreamTech Press
2. Matt Doyle, Beginning PHP 5.3 (wrox-Willey publishing)
3. John Duckett, Beginning HTML, XHTML, CSS and Java script.

Reference Book:

1. HTML, XHTML and CSS Bible, 5ed, Willey India-Steven M. Schafer.

CORE – 11 Practical: Web Technology Lab

1. Acquaintance with elements, tags and basic structure of HTML files.
2. Practicing basic and advanced text for formatting.
3. Practice use of image, video and sound in HTML documents.
4. Designing of web pages- Document layout, list, tables.
5. Practicing Hyperlink of web pages, working with frames.
6. Working with forms and controls.
7. Acquaintance with creating style sheet, CSS properties and styling.
8. Working with background, text, font, list properties.
9. Working with HTML elements box properties in CSS.
10. Develop simple calculator for addition, subtraction, multiplication and division operation using java script.
11. Create HTML page with java script which takes integer number as a input and tells whether the number is odd or even.
12. Create HTML page that contains form with fields name, Email, mobile number, gender, favorite colour and button; now write a java script code to validate each entry. Also write a code to combine and display the information in text box when button is clicked.
13. Write a PHP program to check if number is prime or not.
14. Write a PHP program to print first ten Fibonacci numbers.
15. Create a MySQL data base and connect with PHP.
16. Write PHP script for string and retrieving user information from my SQL table.

a. Write a HTML page which takes Name, Address, Email and Mobile number from user (register PHP).

b. Store this data in MySQL data base.

c. Next page display all user in HTML table using PHP (display .PHP).

17. Using HTML, CSS, Javascript, PHP, MySQL, design a authentication module of a web page.

CORE – 12: Software Engineering

OBJECTIVES:

- To learn the way of developing software with high quality and the relevant techniques.
- To introduce software engineering principles for industry standard.
- To focus on Project management domain and Software risks management.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- An ability to identify, formulates, and solves complex engineering problems by applying principles of engineering, science, and mathematics.
- An ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors.
- An ability to communicate effectively with a range of audiences.
- An ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in global, economic, environmental, and societal contexts.
- An ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives.

Unit-1

Introduction: Evolution of Software to an Engineering Discipline, Software Development Projects, Exploratory Style of Software Development, Emergence of Software Engineering, Changes in Software Development Practices, Computer Systems Engineering. Software Lifecycle Models: Waterfall Model and its Extensions, Rapid Application Development (RAD), Agile Development Models, Spiral Model.

Unit-2

Software Project Management: Software Project Management Complexities, Responsibilities of a Software Project Manager, Project Planning, Metrics for Project Size Estimation, Project Estimation Techniques, Empirical Estimation Techniques, COCOMO, Halstead’s Software Science, Staffing Level Estimation, Scheduling, Organization and Team Structures, Staffing, Risk Management, Software Configuration Management.

Unit-3

Requirement Analysis and Specification: Requirements Gathering and Analysis, Software Requirement Specifications, Formal System Specification Axiomatic Specification, Algebraic Specification, Executable Specification and 4GL.

Software Design: Design Process, Characterize a Good Software Design, Cohesion and Coupling, Layered Arrangements of Modules, Approaches to Software Design (Function Oriented & Object-Oriented).

Unit-4

Coding and Testing: Coding: Code Review, Software Documentation, Testing, Unit Testing, Black Box and White Box Testing, Debugging, Program Analysis Tools, Integration Testing, System Testing, Software Maintenance.

Text Book:

1. Fundamental of Software Engineering, Rajib Mall, Fifth Edition, PHI Publication, India.

Reference Books:

1. Software Engineering– Ian Sommerville, 10/Ed, Pearson.
2. Software Engineering Concepts and Practice – Ugrasen Suman, Cengage Learning India Pvt, Ltd.
3. R. Misra, C. Panigrahi, B. Panda: Principles of Software Engineering & System Design, YesDee Publication

CORE – 12 Practical: Software Engineering Lab

S. No. Practical Title

1. • Problem Statement,
• Process Model
2. Requirement Analysis:
• Creating a Data Flow
• Data Dictionary, Use Cases
3. Project Management:
• Computing FP
• Effort
• Schedule, Risk Table, Timeline chart
4. Design Engineering:
• Architectural Design
• Data Design, Component Level Design
5. Testing:
• Basis Path Testing

Sample Projects:

1. **Criminal Record Management:** Implement a criminal record management system for jailers, police officers and CBI officers.

2. **Route Information**: Online information about the bus routes and their frequency and fares

3. **Car Pooling**: To maintain a web based intranet application that enables the corporate employees within an organization to avail the facility of carpooling effectively.
4. Patient Appointment and Prescription Management System
5. Organized Retail Shopping Management Software
6. Online Hotel Reservation Service System
7. Examination and Result computation system
8. Automatic Internal Assessment System
9. Parking Allocation System
10. Wholesale Management System

CORE–13: Artificial Intelligence

OBJECTIVES:

- To learn the basic concepts of AI principles and approaches.
- To develop the basic understanding of the building blocks of AI.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Compare AI with human intelligence and traditional information processing and discuss its strengths and limitations as well as its application to complex and human-centred problems.
- Discuss the core concepts and algorithms of advanced AI, including informed searching, CSP, logic, uncertain knowledge and reasoning, dynamic Bayesian networks, graphical models, decision making. Multiagent, inductive learning, statistical learning, reinforcement learning, deep learning, natural language processing, robotics, and so on.
- Apply the basic principles, models, and algorithms of AI to recognize, model, and solve problems in the analysis and design of information systems.
- Analyze the structures and algorithms of a selection of techniques related to searching, reasoning, machine learning, and language processing.
- Design AI functions and components involved in intelligent systems such as computer games, expert systems, semantic web, information retrieval, machine translation, mobile robots, decision support systems, and intelligent tutoring systems.
- Review research articles from well-known AI Journals and conference proceedings regarding the theories and applications of AI.
- Carry out a research project and write a research proposal, report and paper.

Unit-1

Introduction to Artificial Intelligence, Background and Applications, Turing Test and Rational Agent approaches to AI, Introduction to Intelligent Agents, their structure, behavior and environment.

Unit-2

Problem Solving and Searching Techniques: Problem Characteristics, Production Systems, Control Strategies, Breadth First Search, Depth First Search, Hill climbing and its Variations, Heuristics Search Techniques: Best First Search, A* algorithm, Constraint Satisfaction Problem, Introduction to Game Playing, Min-Max and Alpha-Beta pruning algorithms.

Unit-3

Knowledge Representation : Introduction to First Order Predicate Logic, Resolution Principle, Unification, Semantic Nets, Conceptual Dependencies, Frames, and Scripts, Production Rules, Conceptual Graphs.

Unit-4

Dealing with Uncertainty and Inconsistencies Truth Maintenance System, Default Reasoning, Probabilistic Reasoning, Bayesian Probabilistic Inference, Possible World Representations, Basics of NLP.

Text books

1. Artificial Intelligence a Modern Approach, Stuart Russell and Peter Norvig,Pearson 3/ed.

Reference books

1. Artificial Intelligence, Rich &Knight , TMG , 3 e/d.
2. DAN.W. Patterson, Introduction to A.I and Expert Systems – PHI, 2007
3. W.F. Clocksin and Mellish, Programming in PROLOG, Narosa Publishing House, 3rd edition, 2001

CORE–13 Practical: Artificial Intelligence Lab

Write a Prolog program

1. To find the factorial of a number
2. To remove the nth item from a list.
3. To find the permutation of a set.
4. To implement append for two lists.
5. To implement palindrome.
6. To find the greater of two numbers X and Y.
7. To find the greatest number in the list of numbers.
8. To find the sum of given list of numbers.
9. To find the reverse of a list.
10. To solve 8 queens problem.
11. To solve 8-puzzle problem using best first search
12. To implement DFS.
13. To implement BFS.

14. To implement best first search.
15. To solve traveling salesman problem.

CORE – 14: Algorithm Design Techniques

OBJECTIVES:

- To be able to learn design principles and concepts of algorithms.
- To have a mathematical foundation in analysis of algorithm.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Argue the correctness of algorithms using inductive proofs and invariants.
- Analyze worst-case running times of algorithms using asymptotic analysis.
- Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize divide-and-conquer algorithms. Derive and solve recurrences describing the performance of divide-and-conquer algorithms.
- Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it.
- Recite algorithms that employ this paradigm. Synthesize dynamic-programming algorithms, and analyze them.
- Describe the greedy paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize greedy algorithms, and analyze them.
- Explain the major graph algorithms and their analyses. Employ Graphs to model engineering problems.

Unit-1

Introduction: Algorithm specification: Pseudo code, Space complexity and time complexity, Analysis and design of Insertion sort algorithm, Divide and Conquer paradigm, Recurrence relations, Solving Recurrences: Substitution methods, Recursion tree method, and Master method.

Unit-2

Searching and Sorting: Analysis of Linear Search, Binary Search, Merge Sort and Quick Sort, Heap Sort. Hashing: Hash functions, Hash table, Collision resolution: Chaining and Open Addressing (Linear probing, Quadratic probing, Double hashing).

Unit-3

Greedy Technique: General Method, Applications: Fractional Knapsack Problem , Job Sequencing with Deadlines, Huffman Codes. **Dynamic Programming:** General Method, Applications: Matrix Chain Multiplication, Longest common subsequence.

Unit-4

Graph Algorithms: Representations of Graphs, Breadth-first search, Depth-first search, Topological sort, Minimum Spanning Trees: Prim's and Kruskal's algorithm, Single-source shortest paths: Bellman-Ford algorithm, Dijkstra's algorithm.

Text books

1. Introduction to Algorithms, by Thomas H, Cormen, Charles E. Leiserson , Ronald L. Rivest, Clifford Stein, PHI.

Reference books

1. Algorithm Desgin, by Jon Kleinberg, Eva Tardos.

CORE – 14 Practical: Algorithm Design Techniques Lab

Using C or C++ implement the following

1. Quick sort.
2. Heap sort.
3. Merge sort.
4. Matrix Multiplication using recursion.
5. Linear Search.
6. Binary Search.
7. Huffman code.
8. Fractional knapsack problem.
9. Matrix chain multiplication.
10. Longest Common Subsequence.
11. Prim's algorithm.
12. Kruskal's algorithm.
13. BFS.
14. DFS.
15. Dijkstra Algorithm.

DSE-1: Numerical Techniques

OBJECTIVES:

- To learn various numerical techniques.
- To be able to implement different numerical techniques using programming language.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Numerical analysis is the design and analysis of techniques to give approximate but accurate solutions to hard problems.
- Advanced numerical methods are essential in making numerical weather prediction feasible.
- Numerical methods give approximate solutions and they are much easier when compared to Analytical methods.
- When analytical solution is impossible, this means that we have to apply numerical methods in order to find the solution.

Unit-1

Floating point representation and computer arithmetic, Significant digits, Errors: Round-off error, Local truncation error, Global truncation error, Order of a method, Convergence and terminal conditions, Efficient computations.

Unit-2

Bisection method, Secant method, Regula−Falsi method Newton−Raphson method, Newton's method for solving nonlinear systems.

Unit-3

Interpolation: Lagrange's form and Newton's form Finite difference operators, Gregory Newton forward and backward differences Interpolation Piecewise polynomial interpolation: Linear interpolation.

Unit-4

Numerical integration: Trapezoid rule, Simpson's rule (only method), Newton−Cotes formulas, Gaussian quadrature, Ordinary differential equation: Euler's method Modified Euler's methods, Runge-Kutta second methods

Text books

1. S.S. Sastry, "Introductory Methods of Numerical Analysis", EEE , 5/ed.
2. M.K. Jain, S.R.K. Iyengar and R.K. Jain, Numerical Methods for Scientific and

Engineering Computation, New Age International Publisher, 6/e (2012)

Reference books

1. Numerical Analysis: J. K. Mantri & S. Prahan, Laxmi Publication.
2. Introduction to Numerical Analysis, Josef Stoer and Roland Bulirsch, Springer.

DSE – 1 Practical: Numerical Techniques Lab

Implement using C/ C++ or MATLAB/ Scilab

1. Find the roots of the equation by bisection method.
2. Find the roots of the equation by secant/Regula−Falsi method.
3. Find the roots of the equation by Newton‘s method.
4. Find the solution of a system of nonlinear equation using Newton’s method.
5. Find the solution of tri-diagonal system using Gauss Thomas method.
6. Find the solution of system of equations using Jacobi/Gauss-Seidel method.
7. Find the cubic spline interpolating function.
8. Evaluate the approximate value of finite integrals using Gaussian/Romberg integration.
9. Solve the boundary value problem using finite difference method.

DSE – 2: Unix Shell Programming

OBJECTIVES:

- To learn the basics of UNIX OS, UNIX commands and File system.
- To familiarize students with the Linux environment.
- To learn fundamentals of shell scripting and shell programming.
- To be able to write simple programs using UNIX.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Identify and use UNIX/Linux utilities to create and manage simple file processing operations, organize directory structures with appropriate security.
- Develop shell scripts to perform more complex tasks.
- Effectively use the UNIX/Linux system to accomplish typical personal, office, technical, and software development tasks.
- Monitor system performance and network activities.
- Effectively use software development tools including libraries, preprocessors, compilers, linkers, and make files.

Unit-1

Introduction: Unix Operating systems, Difference between Unix and other operating systems, Features and Architecture, Installation, Booting and shutdown process, System processes (an overview), External and internal commands, Creation of partitions in OS, Processes and its creation phases – Fork, Exec, wait, exit.

Unit-2

User Management and the File System: Types of Users, Creating users, Granting rights, User management commands, File quota and various file systems available, File System Management and Layout, File permissions, Login process, Managing Disk Quotas, Links (hard links, symbolic links)

Unit-3

Shell introduction and Shell Scripting: Shell and various type of shell, Various editors present in Unix, Different modes of operation in vi editor, Shell script, Writing and executing the shell script, Shell variable (user defined and system variables), System calls, Using system calls, Pipes and Filters.

Unit-4

Unix Control Structures and Utilities: Decision making in Shell Scripts (If else, switch), Loops in shell, Functions, Utility programs (cut, paste, join, tr, uniq utilities), Pattern matching utility (grep).

Text Books:

1. Sumitabha, Das, Unix Concepts And Applications, Tata McGraw-Hill Education, 2017, 4/Ed.

Reference Books:

1. Nemeth Synder & Hein, Linux Administration Handbook, Pearson Education,2010,2/ Ed.

DSE – 2 Practical: Unix Programming Lab

1. Write a shell script to check if the number entered at the command line is prime or not.
2. Write a shell script to modify "cal" command to display calendars of the specified months.
3. Write a shell script to modify "cal" command to display calendars of the specified range of months.
4. Write a shell script to accept a login name. If not a valid login name display message "Entered login name is invalid".
5. Write a shell script to display date in the mm/dd/yy format.
6. Write a shell script to display on the screen sorted output of "who" command along with the total number of users.
7. Write a shell script to display the multiplication table of any number.
8. Write a shell script to compare two files and if found equal asks the user to delete the duplicate file.
9. Write a shell script to find the sum of digits of a given number.
10. Write a shell script to merge the contents of three files, sort the contents and then display them page by page.
11. Write a shell script to find the LCD (least common divisor) of two numbers.
12. Write a shell script to perform the tasks of basic calculator.
13. Write a shell script to find the power of a given number.
14. Write a shell script to find the greatest number among the three numbers.
15. Write a shell script to find the factorial of a given number.
16. Write a shell script to check whether the number is Armstrong or not.

DSE-3: Data Science

OBJECTIVES:

- To learn emerging issues related to various fields of data science.
- To understand the underlying principles of data science, exploring data analysis.
- To learn the basics of R Programming.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Students will develop relevant R programming abilities.
- Students will demonstrate proficiency with statistical analysis of data.
- Students will develop the ability to build and assess data-based models.
- Students will execute statistical analyses with professional statistical software.
- Students will demonstrate skill in data management.
- Students will apply data science concepts and methods to solve problems in real-world contexts and will communicate these solutions effectively.

Unit-1

Data Scientist's Tool Box: Turning data into actionable knowledge, introduction to the tools that will be used in building data analysis software: version control, markdown, git, GitHub, R, and RStudio.

Unit-2

R Programming Basics: Overview of R, R data types and objects, reading and writing data, Control structures, functions, scoping rules, dates and times, Loop functions, debugging tools, Simulation, code profiling.

Unit-3

Getting and Cleaning Data: Obtaining data from the web, from APIs, from databases and from colleagues in various formats, basics of data cleaning and making data "tidy".

Unit-4

Exploratory Data Analysis: Essential exploratory techniques for summarizing data, applied before formal modeling commences, eliminating or sharpening potential hypotheses about the world that can be addressed by the data, common multivariate statistical techniques used to visualize high-dimensional data.

Text Books

1. Rachel Schutt, Cathy O'Neil, "Doing Data Science: Straight Talk from the Frontiline" by
Schroff/O'Reilly, 2013.

Reference Books

1. Foster Provost, Tom Fawcett, “Data Science for Business” What You Need to Know About Data Mining and Data-Analytic Thinking by O'Reilly, 2013.
2. John W. Foreman, “Data Smart: Using data Science to Transform Information into Insight” by John Wiley & Sons, 2013.
3. Eric Seigel, “Predictive Analytics: The Power to Predict who Will Click, Buy, Lie, or Die", 1st Edition, by Wiley, 2013.

DSE-3 Practical: Elementary Data Science Lab

1. Write a program that prints “Hello World” to the screen.
2. Write a program that asks the user for a number n and prints the sum of the numbers 1 to n
3. Write a program that prints a multiplication table for numbers up to 12.
4. Write a function that returns the largest element in a list.
5. Write a function that computes the running total of a list.
6. Write a function that tests whether a string is a palindrome.
7. Implement linear search.
8. Implement binary search.
9. Implement matrices addition, subtraction and Multiplication
10. Fifteen students were enrolled in a course. There ages were:

20 20 20 20 20 21 21 21 22 22 22 22 23 23 23

i. Find the median age of all students under 22 years.

ii. Find the median age of all students.

iii. Find the mean age of all students.

iv. Find the modal age for all students.

v. Two more students enter the class. The age of both students is 23. What is now mean, mode and median?

DSE–4: PROJECT WORK

OBJECTIVE: It is considered as a special course involving application of knowledge in solving / analyzing /exploring a real life situation / difficult problem. A Project/Dissertation work would be of 6 credits. A Project/Dissertation work may be given in lieu of a discipline specific elective paper.

LEARNING OUTCOMES:

After course completion the students will have the following learning outcomes:

- Apply fundamental and disciplinary concepts and methods in ways appropriate to their principal areas of study.
- Demonstrate skill and knowledge of current information and technological tools and techniques specific to the professional field of study.
- Use effectively oral, written and visual communication.
- Identify, analyze, and solve problems creatively through sustained critical investigation.
- Integrate information from multiple sources.
- Demonstrate an awareness and application of appropriate personal, societal, and professional ethical standards.
- Practice the skills, diligence, and commitment to excellence needed to engage in lifelong learning.

GUIDELINES FOR PROJECT FORMULATION

As the project work constitutes a major component in most of the professional programs and it is to be carried out with due care and should be executed with seriousness by the candidates.

TYPE OF PROJECT

As majority of the students are expected to work out a real-life project in some industry/research and development laboratories/educational institutions/software companies, it is suggested that the project is to be chosen which should have some direct relevance in day-to-day activities of the candidates in his/her institution. It is not mandatory for a student to work on a real-life project. The student can formulate a project problem with the help of Guide.

PROJECT PROPOSAL (SYNOPSIS)

The project proposal should be prepared in consultation with the guide. The project proposal should clearly state the project objectives and the environment of the proposed project to be undertaken. The project work should compulsorily include the software development.

The project proposal should contain complete details in the following form:

1. Title of the Project
2. Introduction and Objectives of the Project
3. Project Category (Web Applications/RDBMS/OOPS/Networking/Multimedia/Artificial Intelligence/Data Analysis etc.)
4. Analysis: (DFDs at least up to second level, ER Diagrams/ Class Diagrams/ Database Design etc. as per the project requirements).
5. Design : A complete structure which includes: Number of modules and their description to provide an estimation of the student's effort on the project. Data Structures as per the project requirements for all the modules. Process Logic of each module. Testing process to be used. Reports generation
6. Tools / Platform, Hardware and Software Requirement specifications
7. Future scope and further enhancement of the project.

Mark Distribution/ Evaluation Pattern for Project

Project Report	Presentation	Viva-Voce
60	20	20